Análisis Exploratorio y Curación de Datos

Una perspectiva desde la ingeniería

about.me/edgardohames

Padre y Marido

Computólogo

Cofundador de Bitlogic

Líder de Equipos de Desarrollo

Me gusta enseñar lo que he aprendido :-)

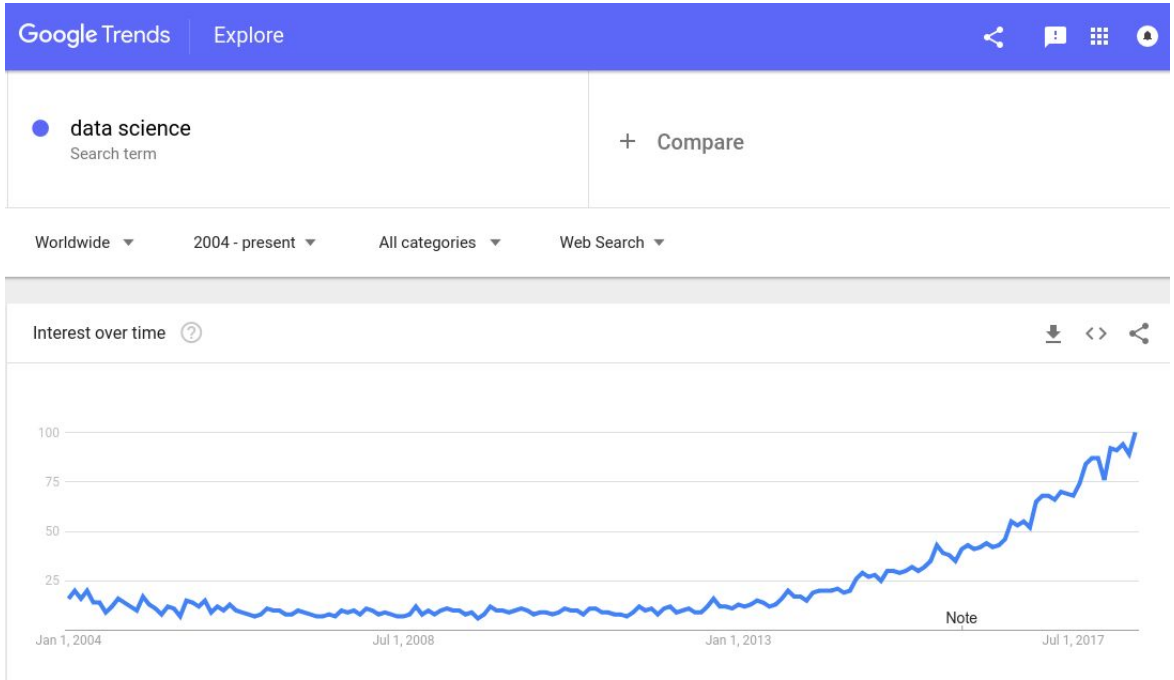
about.me/gmiretti

Formación	Computer Scientist
Misión	as Data Scientist / Engineer,
Experiencia	previous Test Engineer & SQA Manager.
Herramientas:	Using
Agilidad	lean, agile &
Comunidad	open tools
Objetivos:	to make
Técnico	great data products
Moral	and people happier.

Introducción

¿Qué es Ciencia de Datos?

¿Algo nuevo y popular?



¿Qué es Ciencia de Datos?

¿El trabajo más sexy?

Harvard Business Review en 2012

https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century

¿Qué es Ciencia de Datos?

La **ciencia de datos** es un campo interdisciplinario que involucra métodos científicos, procesos y sistemas para extraer conocimiento o un mejor entendimiento de datos en sus diferentes formas, ya sea estructurados o no estructurados

https://es.wikipedia.org/wiki/Ciencia_de_datos

¿Qué es Ciencia de Datos?

https://www.oreilly.com/ideas/what-is-data-science (2010)

La ciencia de datos es la práctica de crear productos de datos.

Un producto de datos es una aplicación que no solo manipula datos sino que obtiene su valor creando información a partir de esos datos.

Productos de Datos

- Google Search
- Sistemas de recomendación de: Amazon, Netflix, Spotify
- Sistemas de publicidad online
- Grammarly
- etc.

¿Cómo hacer productos es Ciencia y no Ingeniería?

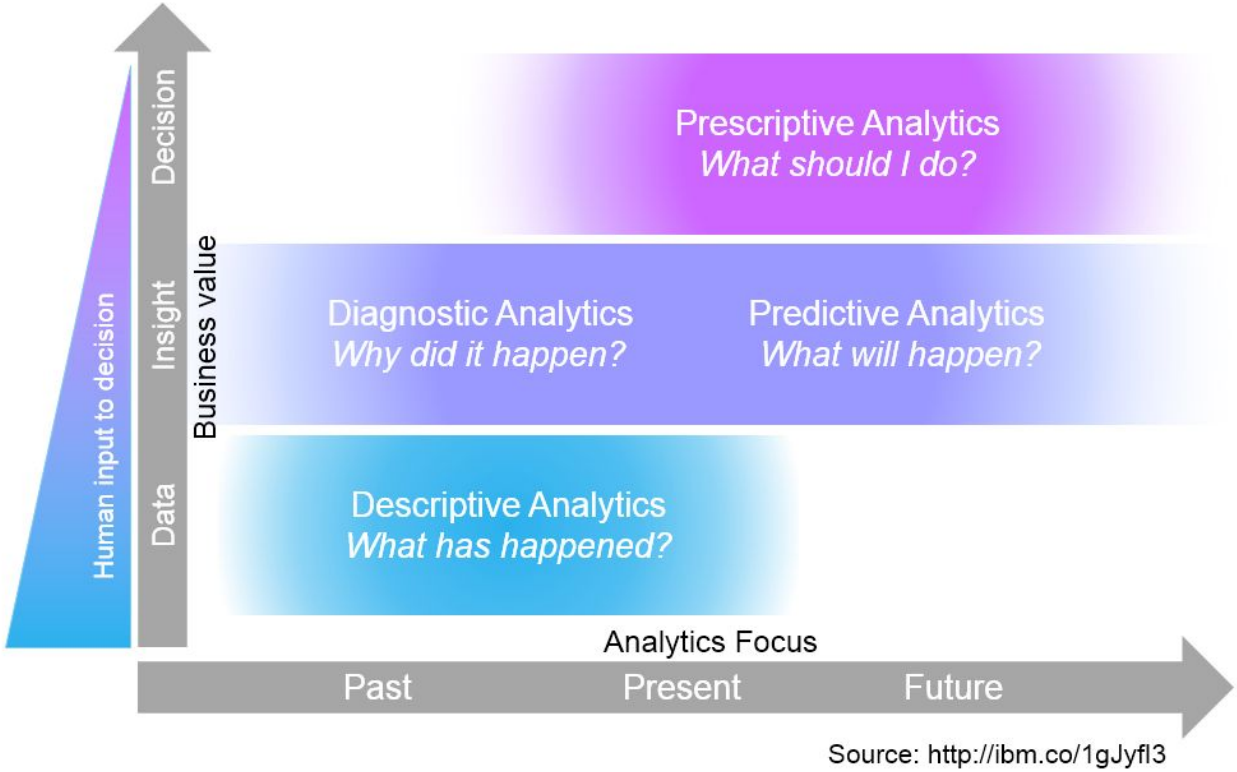
El estado del arte de ingeniería está más cerca que nunca de la ciencia.

Particularmente por el cambio de paradigma del diseño inteligente al descubrimiento de conocimiento, empujado por las metodologías ágiles/lean.

Ciencia de datos es para hacer énfasis en la generación de conocimiento a partir de los datos, proceso propio de las ciencias.

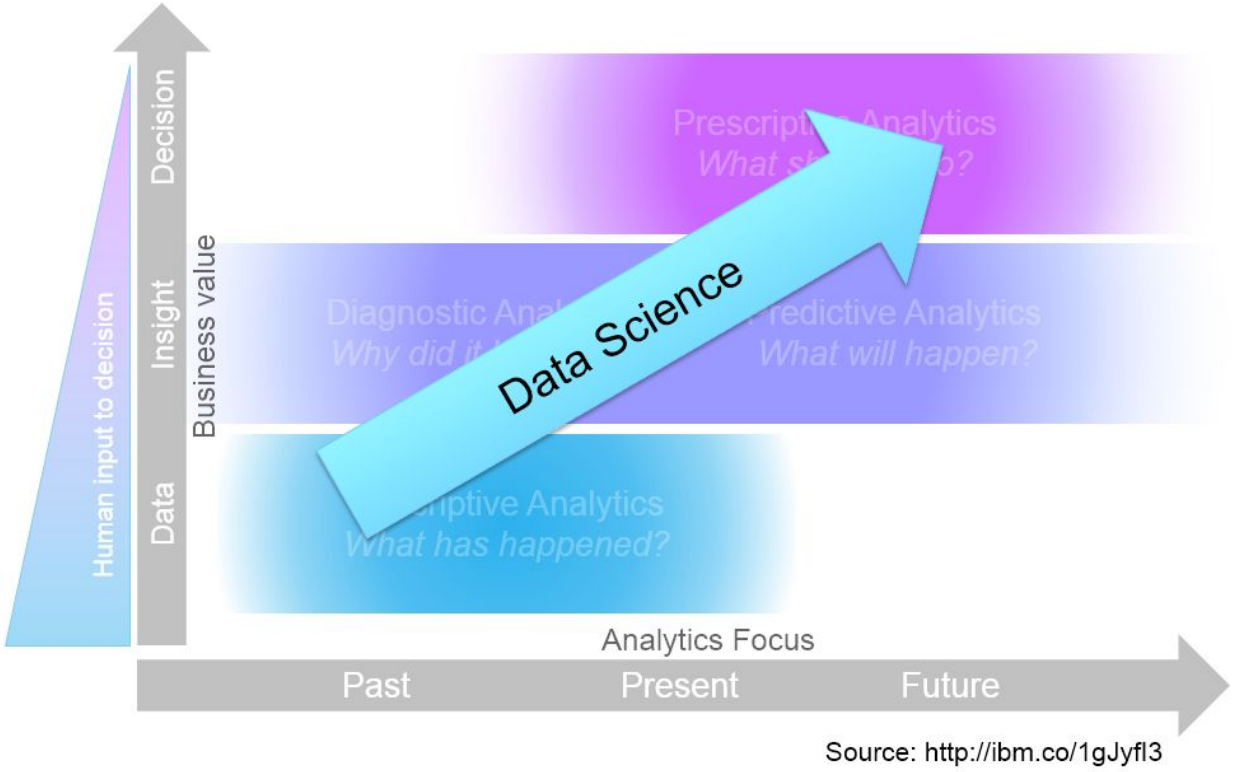
Ingeniería de datos existe y se refiere a las técnicas y prácticas de manipulación de datos, más propias de la ingeniería.

Ciencia de datos y organizaciones

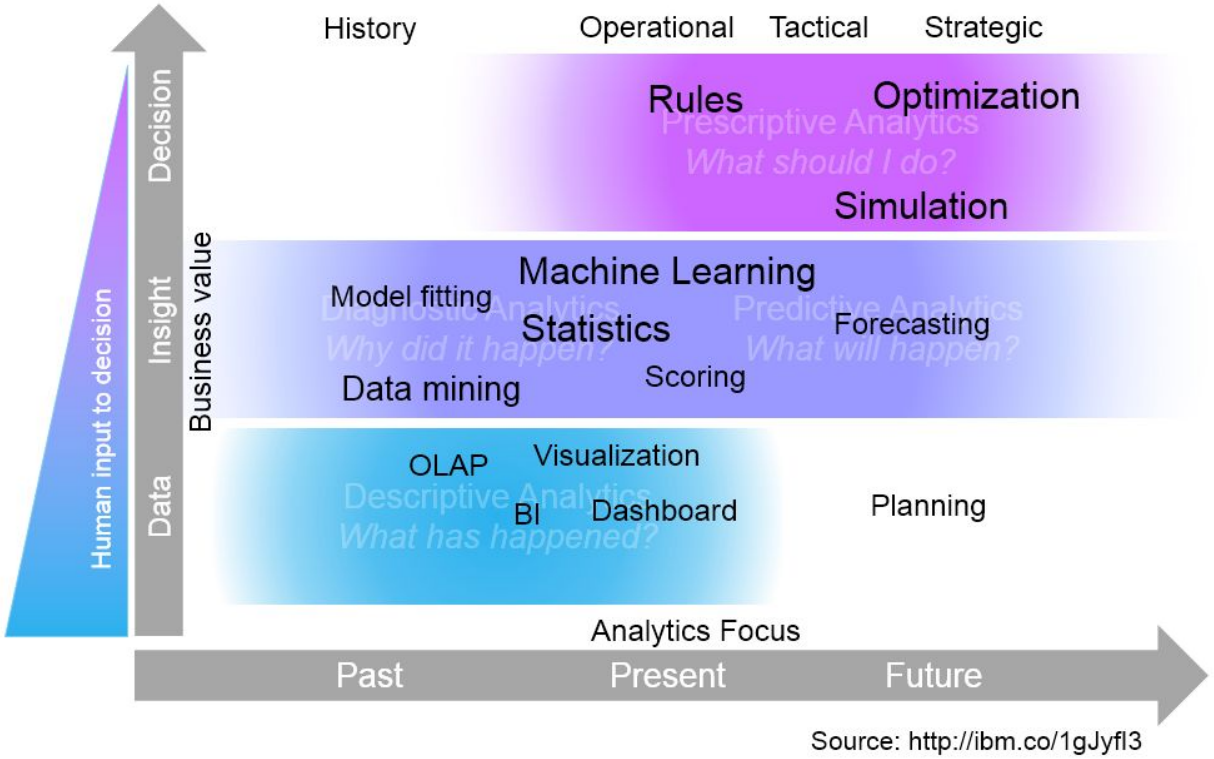


Source: http://ibm.co/1gJyfl3

Ciencia de datos y organizaciones

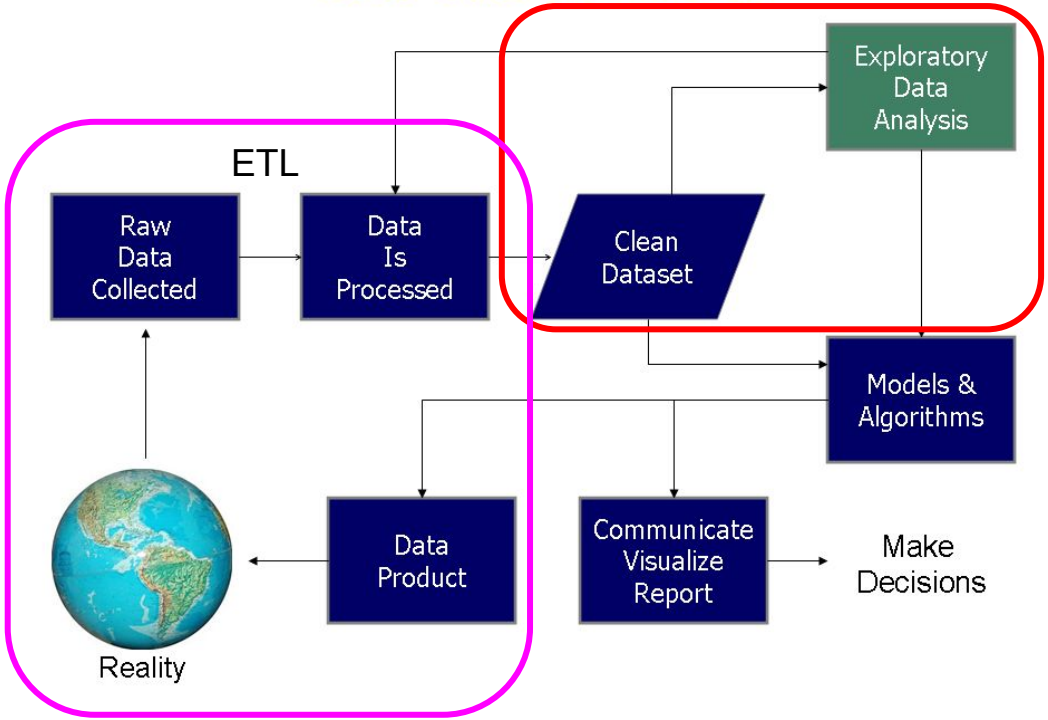


Ciencia de datos y organizaciones

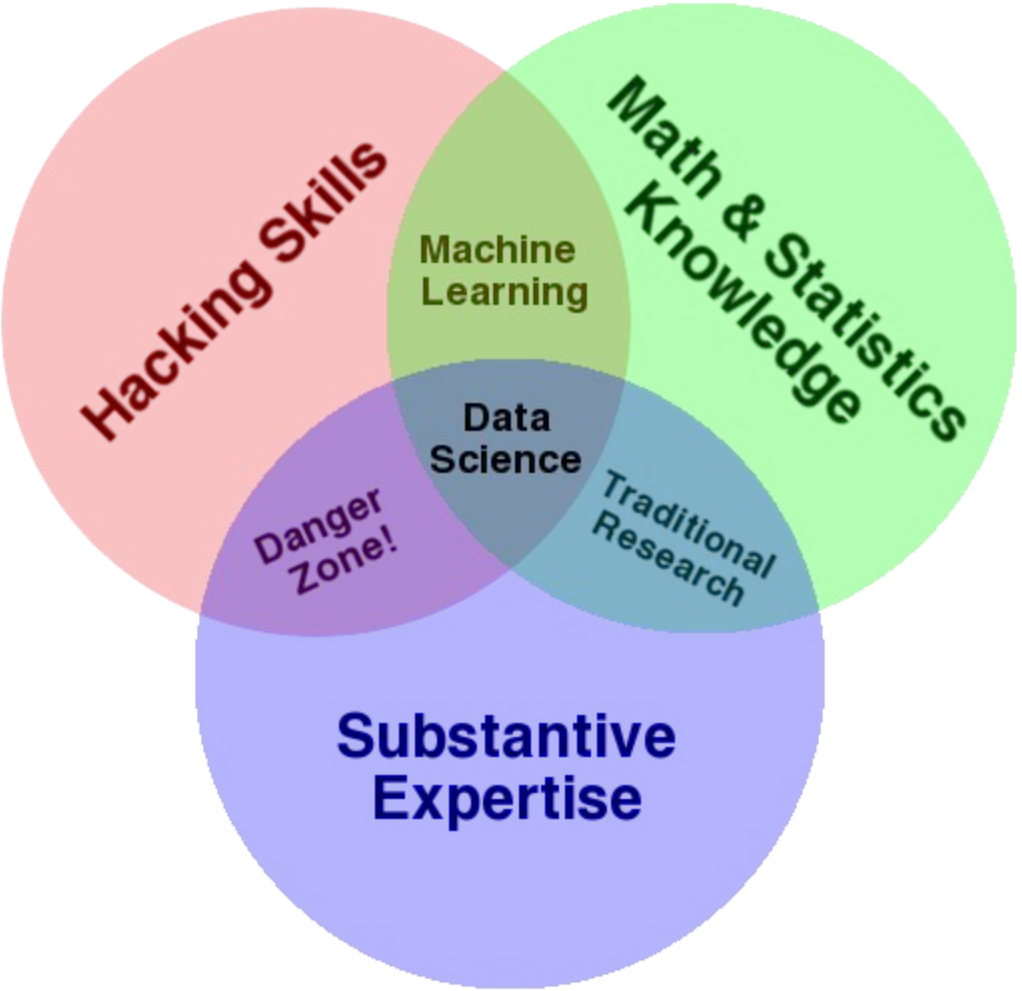


Source: http://ibm.co/1gJyfl3

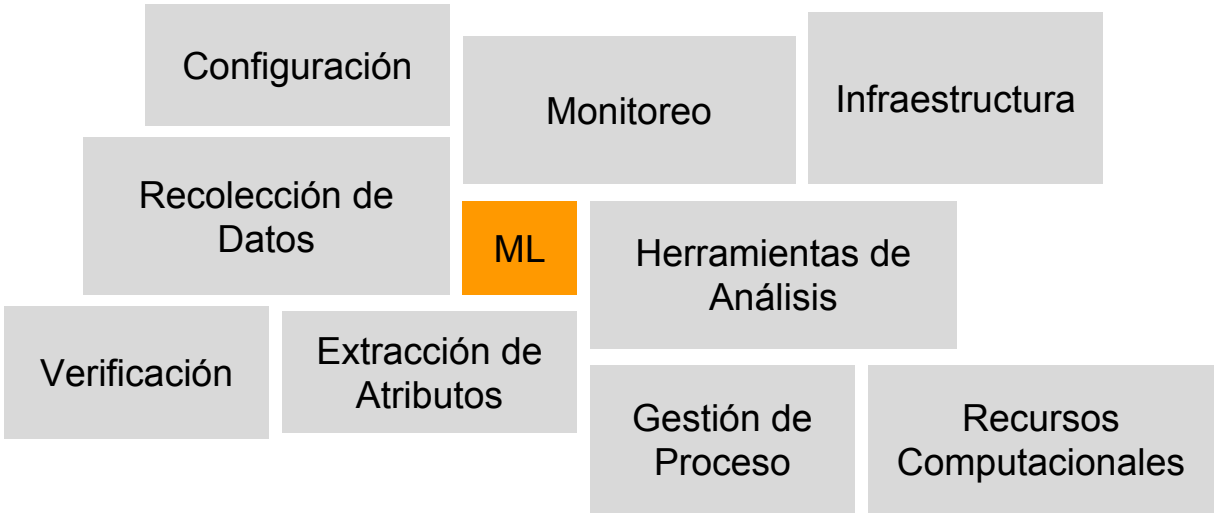
Data Science Process



NOW THIS!

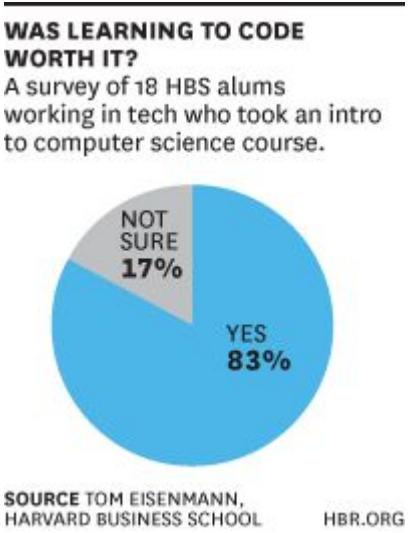

Hacking Skills
Math & Statistics Knowledge
Machine Learning
Data Science
Danger Zone!
Traditional Research
Substantive Expertise

Producto de datos



https://youtu.be/vdG7uKQ2eKk?t=107

¿Por qué aprender a programar?



- Lo decían los MBAs de Harvard en tecnológicas en 2013
  https://hbr.org/2013/09/should-mbas-learn-to-code
  - Hacer software, probablemente no para producción, pero valiosas herramientas internas o prototipos
  - Comunicarse con desarrolladores, para preguntar y entender
  - Contratar
- También lo dicen científicos de datos
  http://treycausey.com/software_dev_skills.html
  - Para colaborar, trabajar en equipo

Pero no todas las organizaciones son tecnológicas...

Por ahora, si aún no hicieron su proceso de transformación digital

Starbucks is becoming a tech company that sells coffee

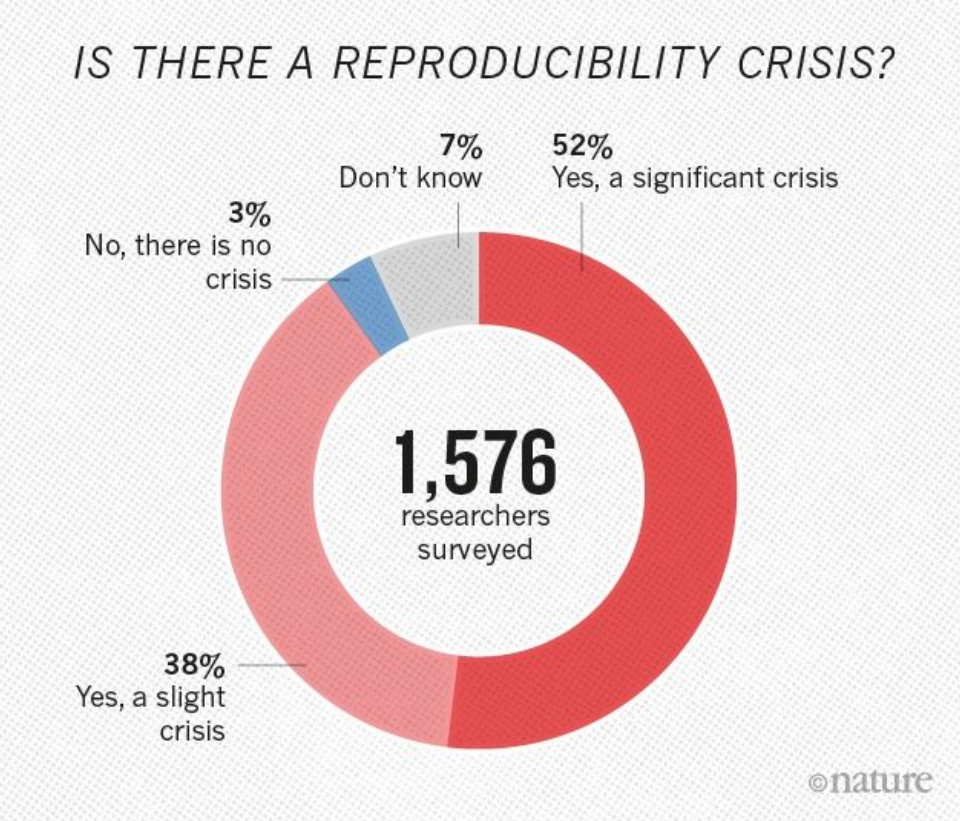
Nicole Sinclair
Markets Correspondent
Yahoo Finance April 28, 2017

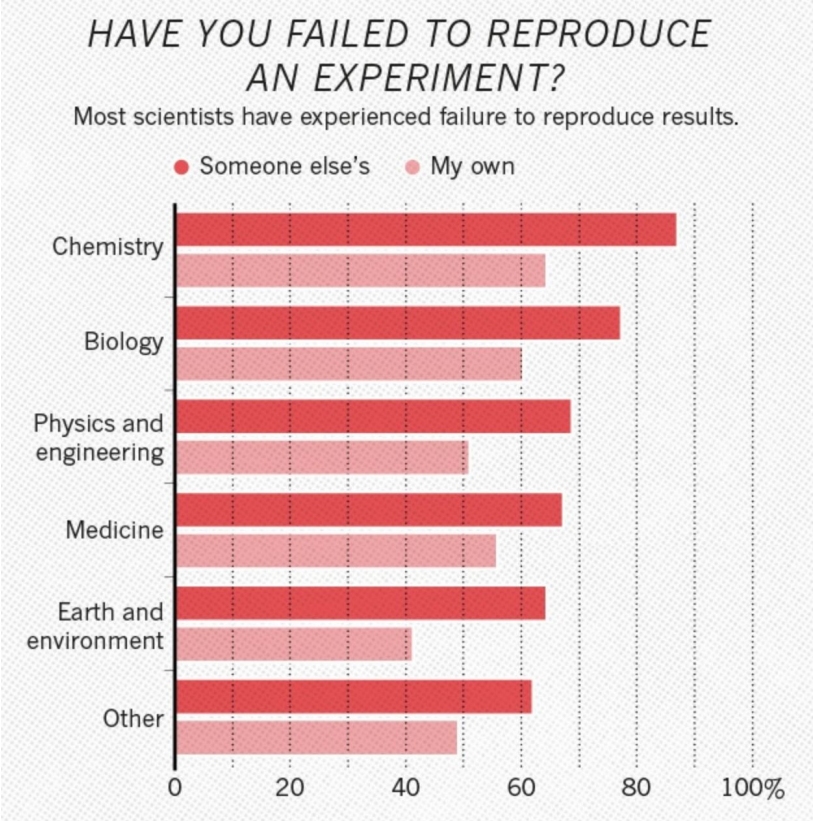
Problema Motivador

Conceptos

Reproducibilidad - capacidad para recomputar resultados (intra-lab).

Repetibilidad - capacidad de otros experimentadores para obtener resultados consistentes (inter-lab)





https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970

Solución a la Crisis de Reproducibilidad

- Disponibilidad de los datos crudos
- Código y documentación para repetir los análisis
- Capacidad de analizar correctamente los datos

http://www.pnas.org/content/112/6/1645.full

Ingesta de Datos

Formatos de Datos

- Tabulares: como una planilla, con filas y columnas.
  - Formatos de Archivos: CSV, TSV, XLS
  - Estructura de Datos: Dataframe
- Jerárquicos: con valores anidados dentro de otros valores.
  - Formatos de Archivos: JSON, XML
  - Estructura de Datos: Lista de Objetos
- Crudos: sin estructura específica
  - Formato de Archivos: TXT
  - Estructura de Datos: String

Formatos: Tabulares vs Jerárquicos vs Crudos

	Sepal.Length	Sepal.Width
1	5.1	3.5
2	4.9	3.0
3	4.7	3.2

Tabular Data

↳Name: Robin
  ↳ Species: Hedgehog
  ↳ Owner: Justice Smith
    ↳ Address: 1234 Main St.
    ↳ Phone #: 123-4567
↳Name: Bunny
  ↳ Species: Rabbit
    ↳ Breed: Holland Lop
  ↳ Color: Brown and white

Hierarchical Data

石室诗士施氏，嗜狮，誓食十狮。氏时时适市视狮。十时，适十狮适市。是时，适施氏适市。氏视是十狮，恃矢势，使是十狮逝世。氏拾是十狮尸，适石室。石室湿，氏使侍拭石室。石室拭，氏始试食是十狮尸。食时，始识是十狮，实十石狮尸。试释是事。

Raw Text

CSV - Comma Separated Values

- Archivos de texto delimitado que usa coma para separar valores.
- Cada línea es un registro con uno o más campos.
- No está formalmente especificado!

```
latitud,longitud,Nombre
-54.832543,-68.3712885,SAN SEBASTIAN  ( USHUAIA )
-54.8249379,-68.3258626,AERO PUBLICO DE USHUAIA
-54.8096728,-68.3114748,PUERTO USHUAIA (PREFECTURA)
-54.8019121,-68.3029511,PUERTO USHUAIA
-51.6896359,-72.2993574,PASO LAURITA CASAS VIEJAS
-51.5866042,-72.3649779,PASO DOROTEA
-51.2544488,-72.2652242,PASO RIO DON GUILLERMO
-53.3229179,-68.6063227,PASO SAN SEBASTIAN
-53.78438,-67.7173342,TERMINAL RIO GRANDE
```

https://tools.ietf.org/html/rfc4180

Lectura de CSV

https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_csv.html

```
In [1]: data = 'col1,col2,col3\na,b,1\na,b,2\nc,d,3'
In [2]: pd.read_csv(StringIO(data))
Out[2]:
```

	col1	col2	col3
0	a	b	1
1	a	b	2
2	c	d	3

Ejercicio: Tablas de Crecimiento

1. Descargar dataset #1 y #2 en CSV
   https://www.cdc.gov/growthcharts/percentile_data_files.htm
2. Graficar las curvas correspondientes a cada percentil para niños y niñas

JSON - JavaScript Object Notation

- Formato estándar que usa texto legible para transmitir objetos en formato clave/valor, arrays o cualquier otro valor serializable.
- Tipos soportados
  - Números: decimales con signo
  - Strings de caracteres Unicode entre “”
  - Booleanos: true/false
  - Array: lista ordenada no tipada entre `[]`
  - Objetos: colección no ordenada clave/valor
  - `null`

```
{
  "firstName": "John",
  "lastName": "Smith",
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "zipCode": "10021"
  },
  "phoneNumber": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "fax",
      "number": "646 555-4567"
    }
  ],
}
```

Lectura de JSON

https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_json.html

```
In: data = [{'state': 'Florida',
             'shortname': 'FL',
             'info': {
                 'governor': 'Rick Scott'
             },
             'counties': [{'name': 'Dade', 'population': 12345},
                          {'name': 'Broward', 'population': 40000},
                          {'name': 'Palm Beach', 'population': 60000}]},
            {'state': 'Ohio',
             'shortname': 'OH',
             'info': {
                 'governor': 'John Kasich'
             },
             'counties': [{'name': 'Summit', 'population': 1234},
                          {'name': 'Cuyahoga', 'population': 1337}]}]
```

```
In: from pandas.io.json import json_normalize

In: json_normalize(data, 'counties', ['state', 'shortname',
['info', 'governor']])
```

Out:

	name	population	state	shortname	info.governor
0	Dade	12345	Florida	FL	Rick Scott
1	Broward	40000	Florida	FL	Rick Scott
2	Palm Beach	60000	Florida	FL	Rick Scott
3	Summit	1234	Ohio	OH	John Kasich
4	Cuyahoga	1337	Ohio	OH	John Kasich

Ejercicio: Pasos Fronterizos de Argentina

Leer ubicaciones de pasos fronterizos y dibujar en un mapa.

https://github.com/vgm64/gmplot

Ejercicio: Analizar texto

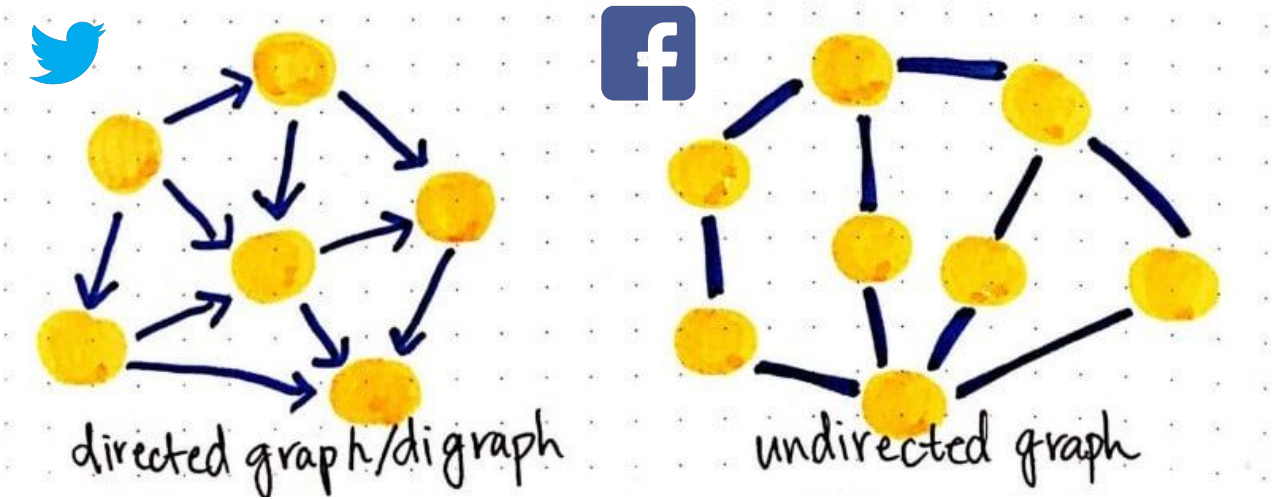
- Utilizar un texto de proyecto Gutenberg en castellano
  http://www.gutenberg.org/browse/languages/es
- Contar palabras y ordenar por frecuencia
  - Limpiar preludio y licencia de Project Gutenberg
  - Omitir “palabras vacías” (stop words) y símbolos
- Encontrar personajes
- Hacer un análisis extra a gusto
- Hint: Tutorial para español
  https://relopezbriega.github.io/blog/2017/09/23/procesamiento-del-lenguaje-natural-con-python/
  - Usa spacy.io y github.com/chartbeat-labs/textacy
  - No hace falta la parte de Deep Learning

Grafos: Conceptos

Grafo: Un par ordenado de **vértices** y **aristas** entre vértices.

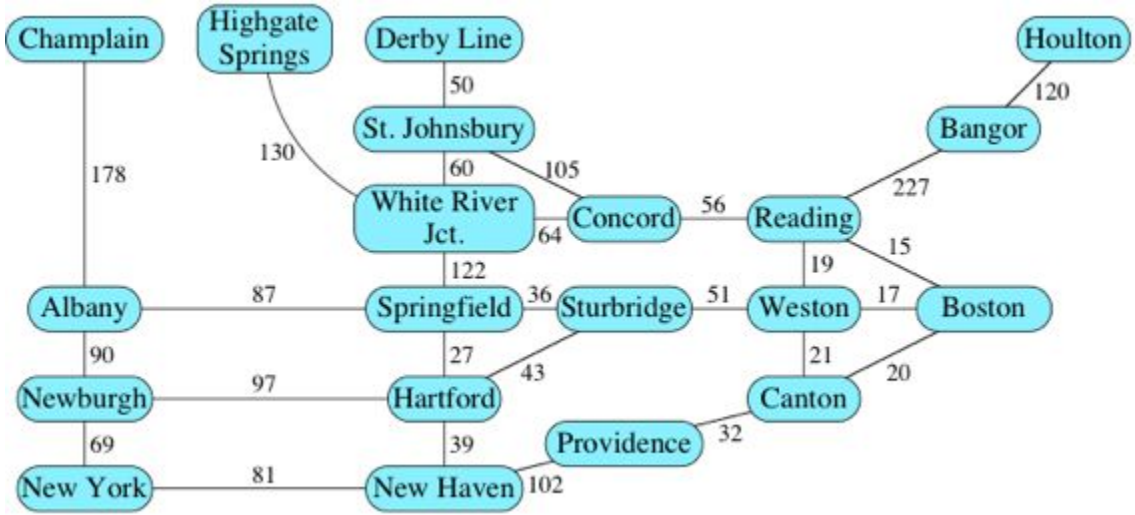
Grafo Dirigido: La arista es un par ordenado y tiene dirección: (a,b) distinto de (b,a)

Grafo No Dirigido: La arista es un conjunto y no tiene dirección: {a,b} igual a {b,a}

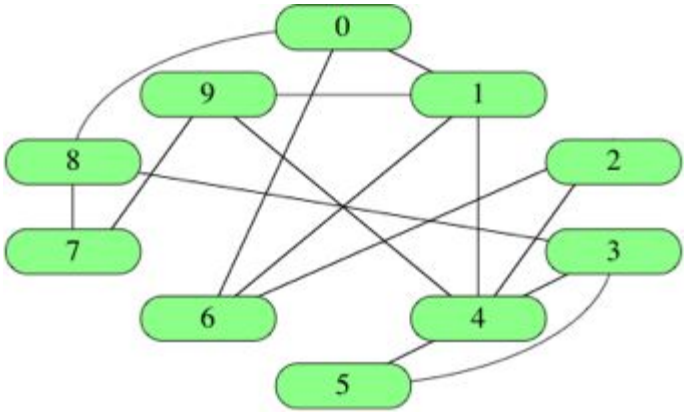


Grafos con Pesos

Tiene una métrica asociada a cada arista.

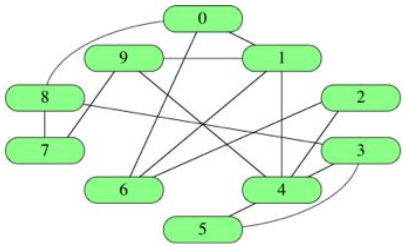


Grafo como lista de aristas




```
[ [0,1], [0,6], [0,8], [1,4],
[1,6], [1,9], [2,4], [2,6], [3,4],
[3,5], [3,8], [4,5], [4,9], [7,8],
[7,9] ]
```

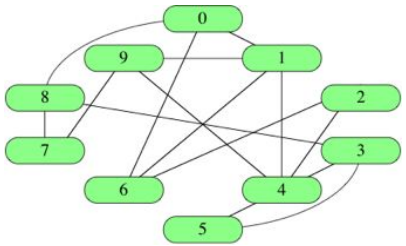
Grafo como matriz de adyacencia

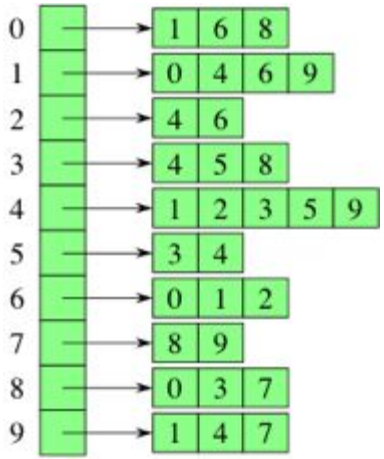


	0	1	2	3	4	5	6	7	8	9
0	0	1	0	0	0	0	1	0	1	0
1	1	0	0	0	1	0	1	0	0	1
2	0	0	0	0	1	0	1	0	0	0
3	0	0	0	0	1	1	0	0	1	0
4	0	1	1	1	0	1	0	0	0	1
5	0	0	0	1	1	0	0	0	0	0
6	1	1	1	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	1	1
8	1	0	0	1	0	0	0	1	0	0
9	0	1	0	0	1	0	0	1	0	0

```
[ [0, 1, 0, 0, 0, 0, 1, 0, 1, 0],
  [1, 0, 0, 0, 1, 0, 1, 0, 0, 1],
  [0, 0, 0, 0, 1, 0, 1, 0, 0, 0],
  [0, 0, 0, 0, 1, 1, 0, 0, 1, 0],
  [0, 1, 1, 1, 0, 1, 0, 0, 0, 1],
  [0, 0, 0, 1, 1, 0, 0, 0, 0, 0],
  [1, 1, 1, 0, 0, 0, 0, 0, 0, 0],
  [0, 0, 0, 0, 0, 0, 0, 0, 1, 1],
  [1, 0, 0, 1, 0, 0, 0, 1, 0, 0],
  [0, 1, 0, 0, 1, 0, 0, 1, 0, 0]
]
```

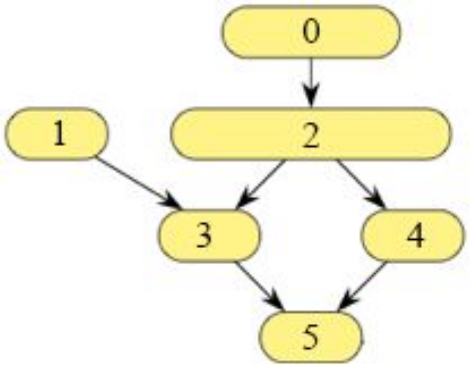
Grafo como lista de adyacencia






```
[ [1, 6, 8],
  [0, 4, 6, 9],
  [4, 6],
  [4, 5, 8],
  [1, 2, 3, 5, 9],
  [3, 4],
  [0, 1, 2],
  [8, 9],
  [0, 3, 7],
  [1, 4, 7] ]
```

Ejercicio: Representación de grafos




```
edgeList = [ [0, 2], [1, 3], [2, 3],
  [2, 4], [3, 5], [4, 5] ];

adjMatrix = [...];

adjList = [...];
```

Lectura Complementaria

Se recomienda revisar el artículo “Nociones sobre Grafos” del repo.

Notebooks sobre análisis de grafos de redes sociales en Python de la EAIA 2017
https://bitbucket.org/gmiretti/social-network-analysis

Bases de Datos

Una base de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

Clasificadas según

- Variabilidad de los datos: estáticas o dinámicas
- Contenidos: bibliográficas, texto completo, directorios, etc
- Modelo de administración: [no] transaccionales, [no] relacionales, distribuidas, etc

Bases de Datos Relacionales

- Basada en tuplas (listas ordenadas de elementos).
- Se compone de varias tablas (relaciones)
- Cada tabla es un conjunto de campos (columnas) y registros (filas)
- Se establecen relaciones entre tablas usando claves
- Permiten combinar columnas de una o más tablas (JOIN)

Modelo Relacional: Ejemplo

Modelo relacional

Ejemplo de datos

jugadores

id_jugador	nombre	id_equipo
1	C. Ronaldo	1
2	Messi	2
3	Griezmann	3
4	Luis Suárez	2
5	Bale	1

Clave secundaria
FK

equipos

id_equipo	nombre
1	Real Madrid
2	F.C. Barcelona
3	Atlético de Madrid

Clave primaria
PK

ACID

ACID - **A**tomicity, **C**onsistency, **I**solation, **D**urability

Atomicidad: Serie de operaciones indivisible e irreducibiles que ocurren todas juntas o ninguna.

Consistencia: Cualquier transacción que comienza en el futuro verá los efectos de las transacciones que ocurrieron en el pasado.

Aislamiento: Visibilidad de la integridad de una transacción para otros usuarios y sistemas (manejo de concurrencia).

Durabilidad: transacciones completadas sobrevivirán de manera permanente.

Bases de Datos No Relacionales

- No permiten JOIN
- No intentan garantizar ACID
- Escalan horizontalmente (agregación de 2 o más instancias de DB)

Bases de Datos Documentales

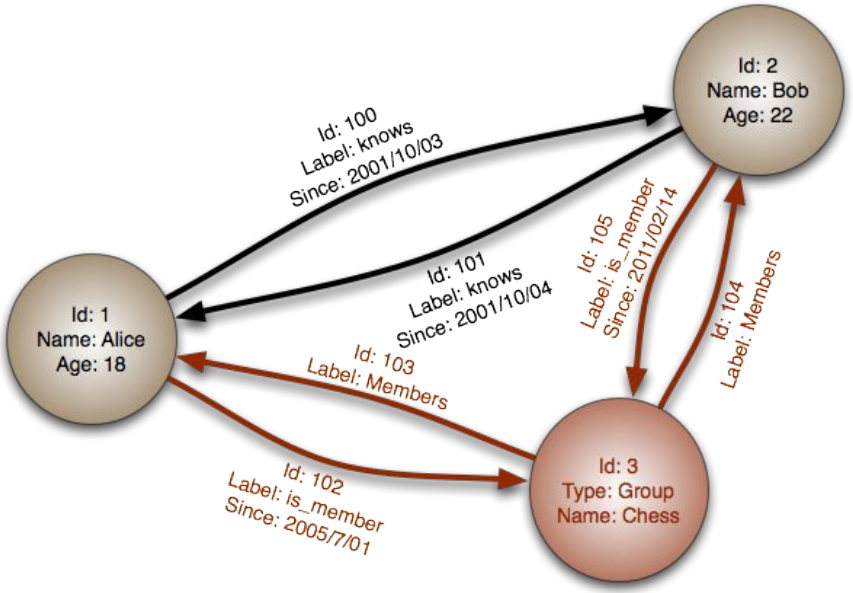
- Almacenan, Recuperan y Gestionan “documentos” (datos estructurados, pero no necesariamente todos iguales)
- Son más fáciles de extender: se pueden agregar o quitar campos a los documentos. No tienen la idea de “campos vacíos”.

```
{
    _id: 1234,
    Nombre:"Pepe",
    Dirección:"Plaza Mayor 5",
    Profesión:"Panadero"
}
```

```
{
    _id: 4567,
    Nombre:"Juan",
    Dirección:"Vía Azul 13",
    Hijos:[
        {Nombre: “Miguel”, Edad: 3},
        {Nombre: “Sara”, Edad: 5},
    ]
}
```

Bases de Datos Orientadas a Grafos

- Representan la información como nodos y relaciones en un grafo.
- Son ideales para almacenar información jerárquica y que debe ser consultada o recorrida como tal.
- Implementan nativamente algoritmos o consultas propias de grafos (por ejemplo, camino más corto entre nodos)

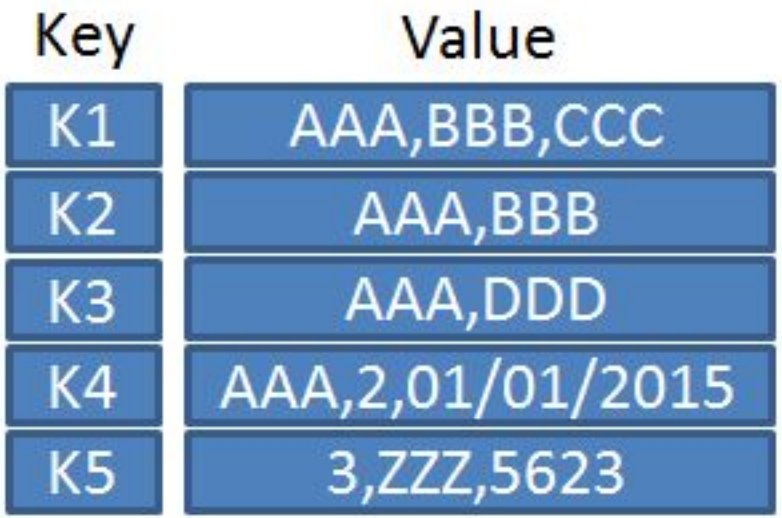


Bases de Datos Orientadas a Objetos

- Representan la información como objetos de la programación orientada a objetos (añaden persistencia a los objetos de manera transparente).
- Acceden a los datos usando los punteros (sin búsquedas)

Bases de Datos Clave/Valor

Gestionan arrays asociativos, (diccionarios o mapas). Los valores son “opacos” y su estructura puede ser completamente diferente entre sí.



Solución a la Crisis de Reproducibilidad

- Disponibilidad de los datos crudos ✓
- Código y documentación para repetir los análisis
- Capacidad de analizar correctamente los datos

http://www.pnas.org/content/112/6/1645.full

Código Reproducible

Código Reproducible: Problemas

- Compilación o instalación fallidas por falta de dependencias o documentación incorrecta
- Evolución/Erosión de Software
- Barreras para la adopción y el reuso

ftp://ftp.cs.arizona.edu/ reports/2013/TR13-03.pdf
http://storify.com/hlapp/reproducibility-repeatability-bigthink

Código Reproducible: Problemas

- Compilación o instalación fallidas por falta de dependencias o documentación incorrecta
- Evolución/Erosión de Software
- Barreras para la adopción y el reuso

COMUNES AL DESARROLLO DE SOFTWARE

ftp://ftp.cs.arizona.edu/ reports/2013/TR13-03.pdf
http://storify.com/hlapp/reproducibility-repeatability-bigthink

Gestión de Configuración

Un **proceso** de ingeniería de sistemas para establecer y mantener la **consistencia** de los **atributos físicos y funcionales** de un producto con sus requerimientos, diseño e información operacional durante todo su ciclo de vida.

Gestión de Configuración de Software

Es la tarea de seguimiento y control de cambios en el software.

Incluye control de revisiones y establecimiento de líneas base.

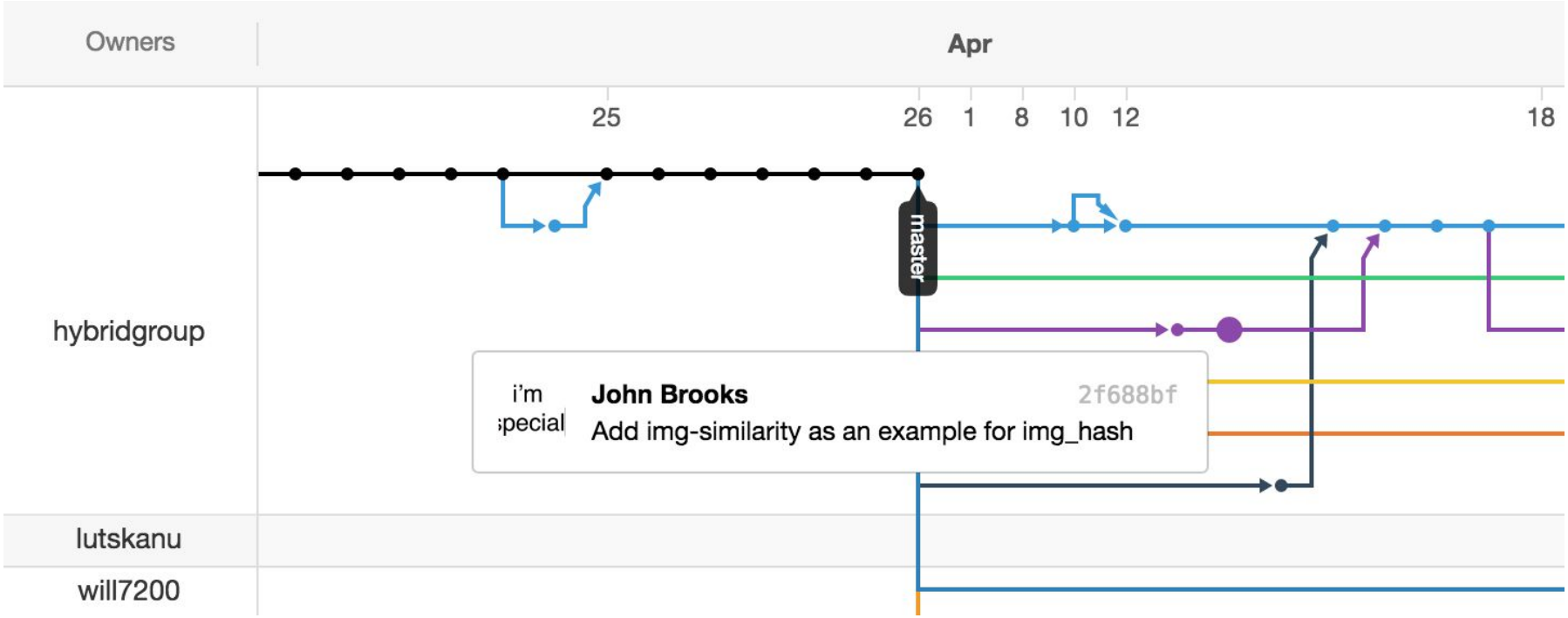
Permite determinar quién hizo qué cambio y cómo replicar dichos cambios en distintos entornos.

Gestión de Código: git

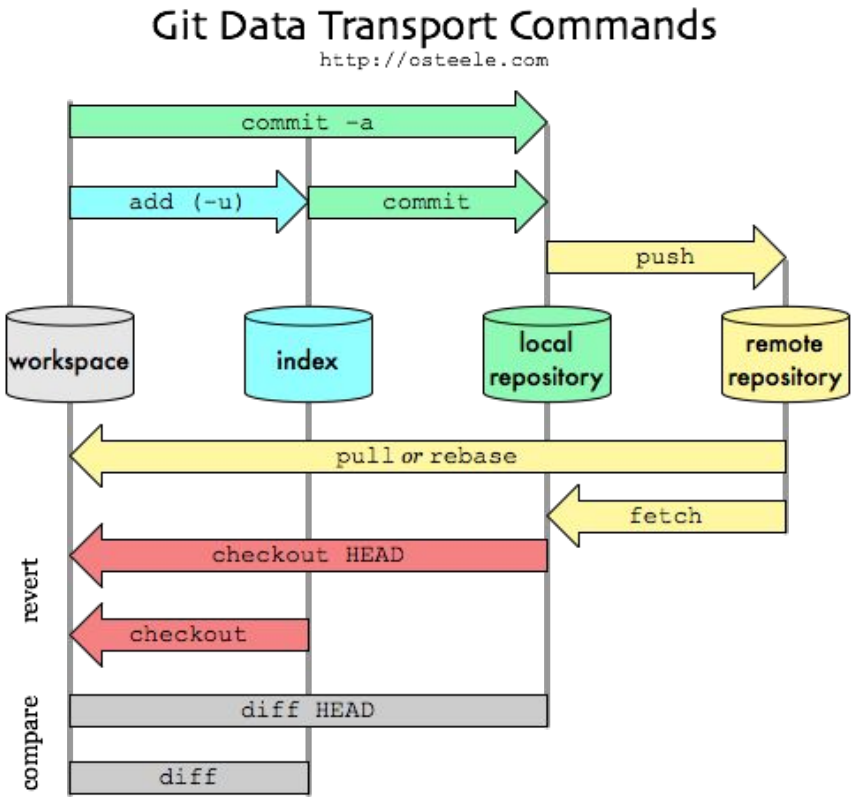
Sistema de control de versiones para seguimiento de cambios en archivos de computadora y coordinación de trabajo entre múltiples personas.

Es distribuido y tiene como objetivos la velocidad, la integridad de datos y flujos de trabajo distribuidos y no lineales.

Ejemplo de flujo no lineal



Git Resumen de Comandos



Recomendación: Trunk Based Development

Usar un esquema de colaboración con una única rama llamada *trunk* o *master* y resistir la presión (o tentación) de crear ramas de colaboración longevas.

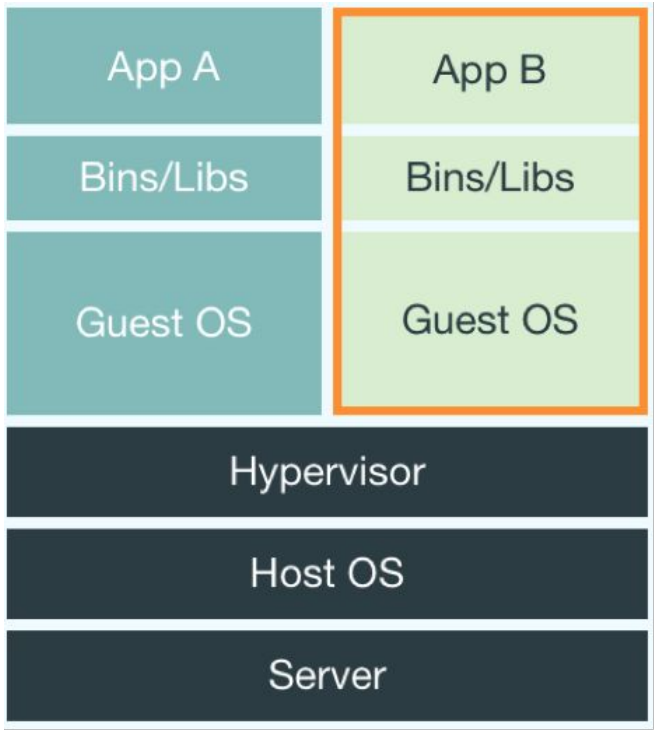
https://trunkbaseddevelopment.com/

Así se evita el infierno de integración, los builds rotos y se vive feliz para siempre :-)

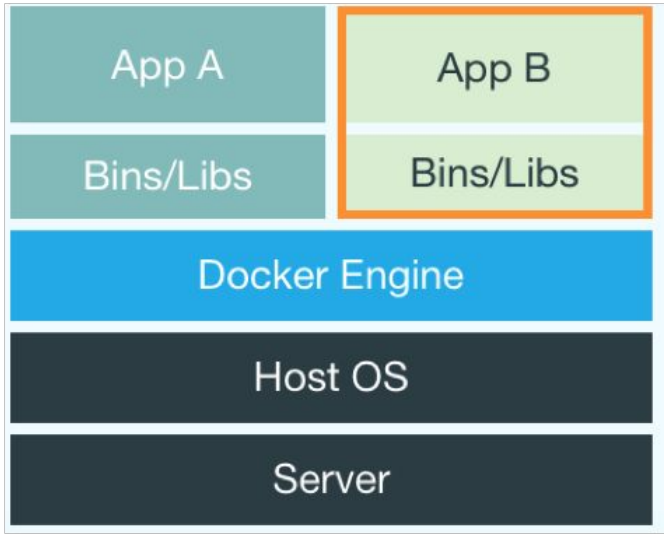
Distribución: Docker

- Es un plataforma para empaquetar software en unidades estandarizadas para desarrollo, distribución y despliegue.
- Encierra un sistema de archivos completo que incluye todo lo necesario para ejecutar un proceso: código, entorno de ejecución, y herramientas y bibliotecas del sistema.
- Garantiza que su ejecución será siempre igual en cualquier entorno.
- Realiza virtualización a nivel de sistema operativo (no confundir con máquinas virtuales).

Comparación VM vs Contenedores Docker



VM



Docker

Docker: Conceptos

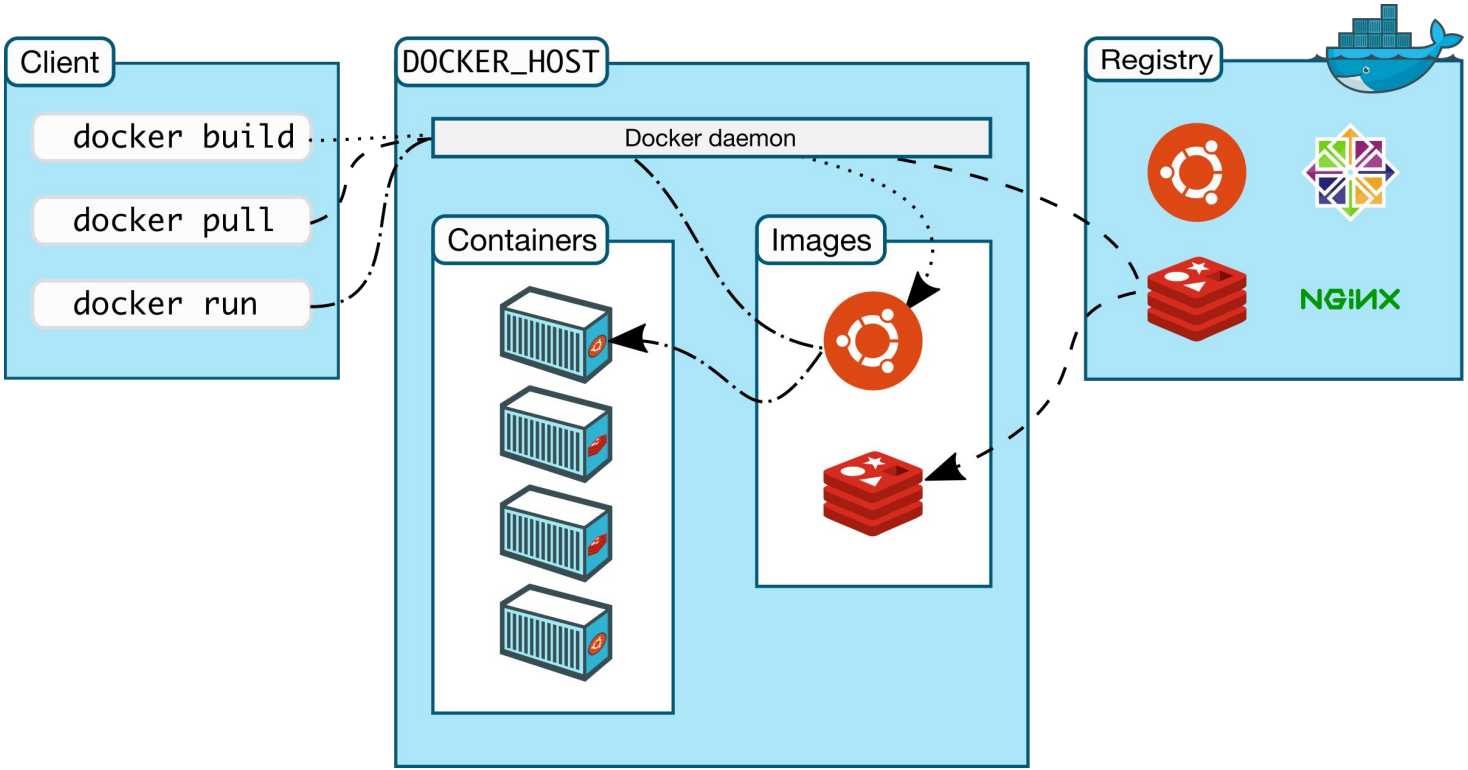
Imagen: colección de cambios a un sistema de archivos y parámetros para ejecución como contenedor

Contenedor: instancia en ejecución de una imagen.

Dockerfile: archivo con instrucciones para construir una imagen.

Registry: repositorio de imágenes de Docker.

Arquitectura de Docker



Docker: Comandos básicos

`build`: Construir una imagen

`tag`: Etiquetar una imagen

`push`: Subir una imagen a un repositorio

`pull`: Descargar una imagen

`run`: Ejecutar un contenedor

`stop`: Detener un contenedor

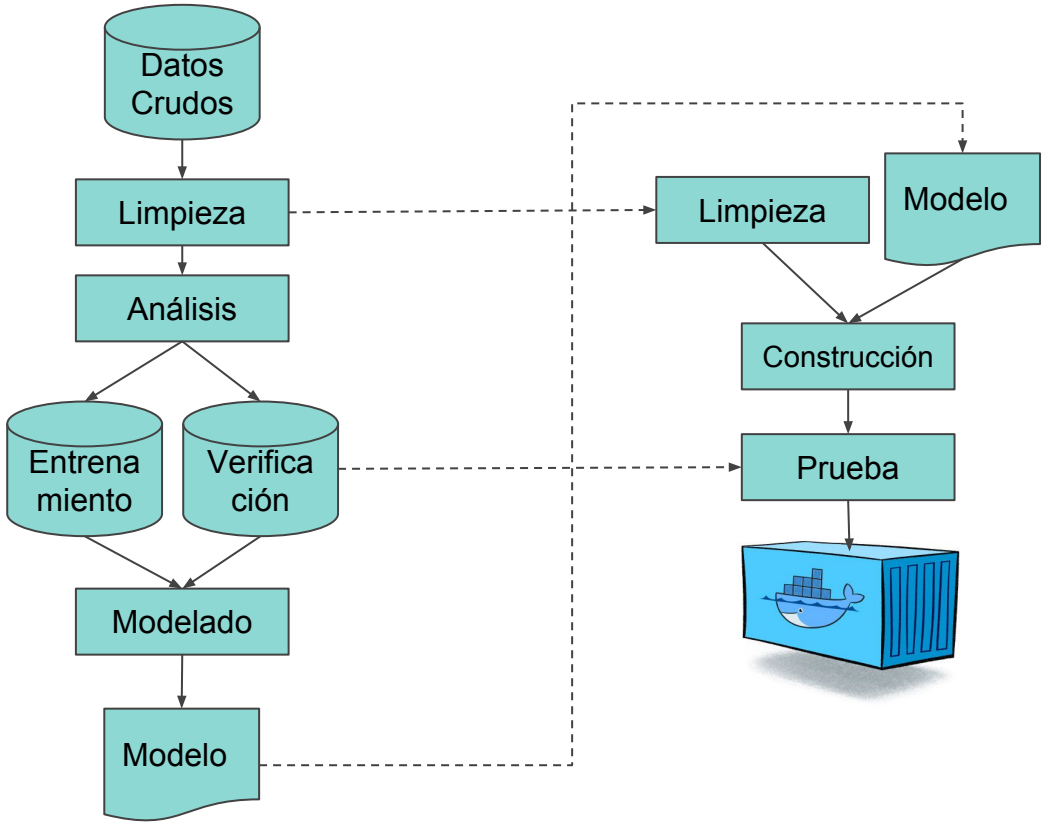
`ps`: Consultar el estados de los contenedores

`rm`: Remover un contenedor

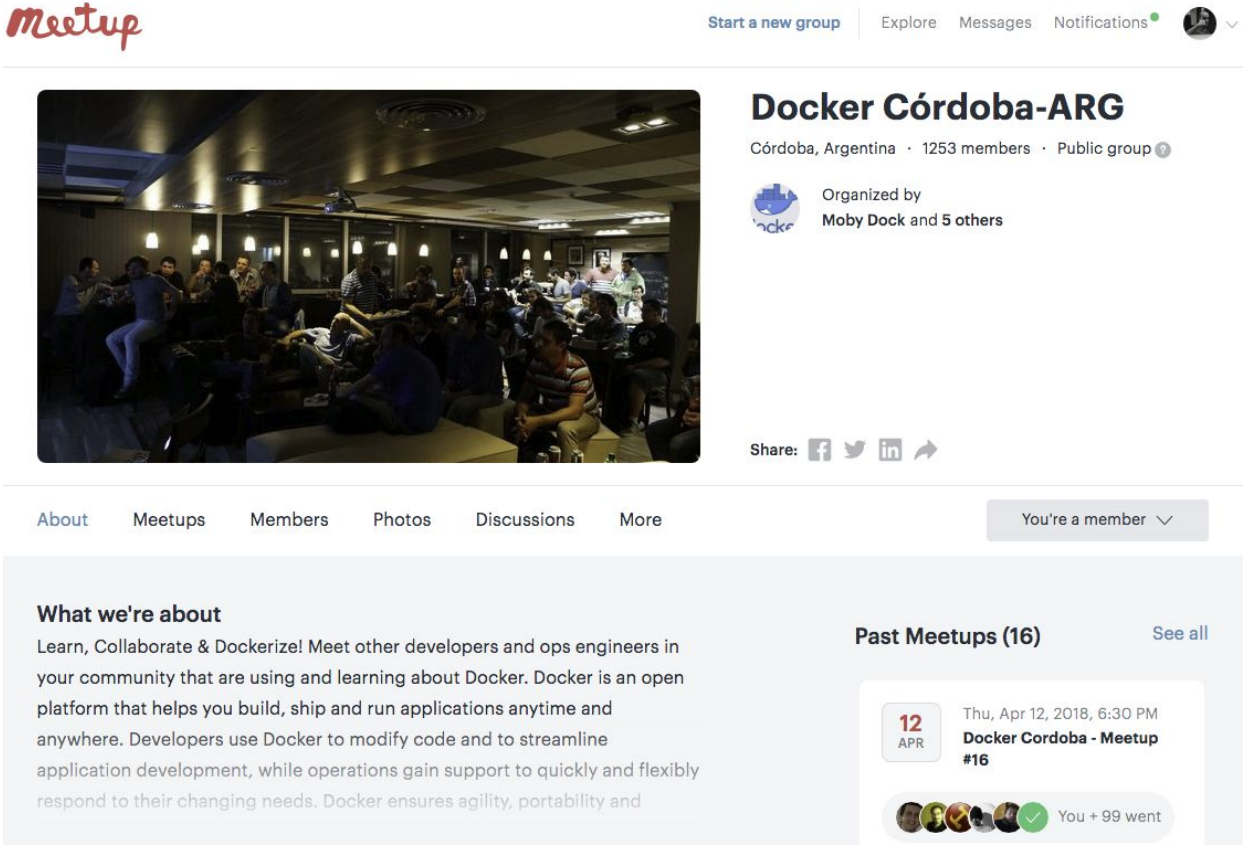
Docker: Beneficios

- Imágenes inmutables con ejecuciones reproducibles.
- Mecanismo de ejecución estándar y unificado.
- Despliegues simplificados e independientes del entorno.

ML vs Software Pipeline



Recomendación: Meetup de Docker Córdoba



Cursos Complementarios

Essential Skills for reproducible Research Computing
https://barbagroup.github.io/essential_skills_RRC/

Reproducible Analysis and Research Transparency
https://reproducible-analysis-workshop.readthedocs.io/en/latest/

Solución a la Crisis de Reproducibilidad

- Disponibilidad de los datos crudos ✓
- Código y documentación para repetir los análisis ✓
- Capacidad de analizar correctamente los datos

http://www.pnas.org/content/112/6/1645.full

Análisis y Curación

Limpiando datos

El Banco Mundial tiene experiencia en análisis de datos y lineamientos

https://dimewiki.worldbank.org/wiki/Data_Cleaning

En particular, vamos a revisar su checklist

https://dimewiki.worldbank.org/wiki/Checklist:_Data_Cleaning

No tanta ciencia, sino experiencia, heurísticas y prácticas

Limpiando datos

1. Importando los datos
    1.1. Verificar si no hay problemas en la importación
    Habilitar chequeos al importar
    1.2. Asegurar de tener ids/claves únicas
    Chequear que no hay datos duplicados
    1.3. Despersonalizar datos y guardarlos en un nuevo archivo
    AR https://www.argentina.gob.ar/aaip/datospersonales/derechos EU **GDPR**
    1.4. Nunca modificar los datos crudos u originales

Limpiando datos

2. Pasos necesarios

2.1. Etiquetas de variables/columnas: no usar caracteres especiales
Verificar que no haya problemas de codificación/encoding

2.2. Tratar valores faltantes
Quitar o imputar

2.3. Codificar variables: las variables categóricas deben ser etiquetadas como variables numéricas, no como cadenas

2.4. No cambiar los nombres de las variables de la fuente de origen

2.5. Verificar la consistencia de las variables
Aplicar reglas de integridad

2.6. Identificar y documentar valores atípicos/outliers
Calcular estadísticos

2.7. Evaluar cómo comprimir los datos para su almacenamiento más eficiente

2.8. Guardar el set de datos con un nombre informativo.

Limpiando datos

3. Pasos deseables

3.1. Ordenar variables/columnas si es posible – primero ID, luego en el mismo orden que la fuente

3.2. Quitar variables/columnas que no tienen información a analizar

3.3. Renombrar variables de grillas

3.4. Categorizar resultados en “Otros”
Si tiene un campo de texto libre asociado, codificar en nuevos valores de la variable categórica asociada. Revisar fuzzyness.

3.5. Agregar metadata a los datos: cuando y como fueron obtenidos, limpieza realizada, asunciones, etc
Vincular con etiquetas del código fuente y los datos. Al menos incluir un README.

Limpiando datos - Práctico en Python+scipy stack

jupyter.org

python

ANACONDA

NumPy

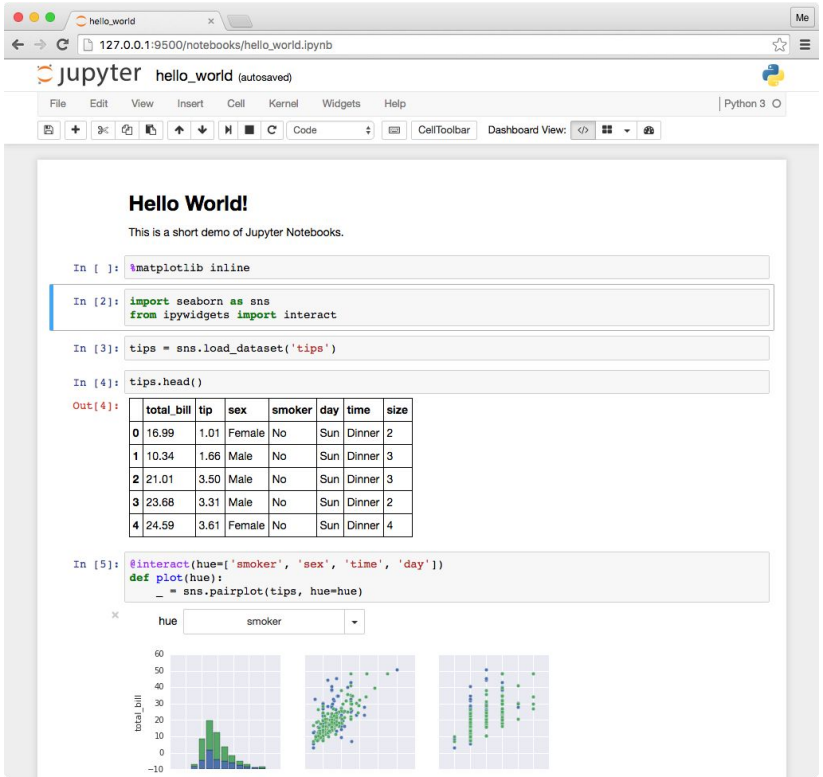
SciPy

matplotlib

IP[y]: IPython
Interactive Computing

pandas

scikit learn



Limpiando datos - Práctico

https://github.com/DiploDatos/AnalisisYCuracion

Limpiando datos - Práctico usando docker (próximamente)

```
$ docker run -it --rm -v $PWD:/home/jovyan/work -e NB_UID=`id -u` -p 8888:8888 jupyter/scipy-notebook
```

- Esperan a que la imagen se descargue
  - Abren con el navegador
  - Abren notebook y tienen disponible:
    - Jupyter Notebook 5.2.x
    - Conda Python 3.x environment
    - Scipy Stack pre-installed: pandas, matplotlib, scipy, seaborn, scikit-learn, scikit-image, sympy, cython, patsy, statsmodel, cloudpickle, dill, numba, bokeh, vincent, beautifulsoup, xlrd

Lectura complementaria

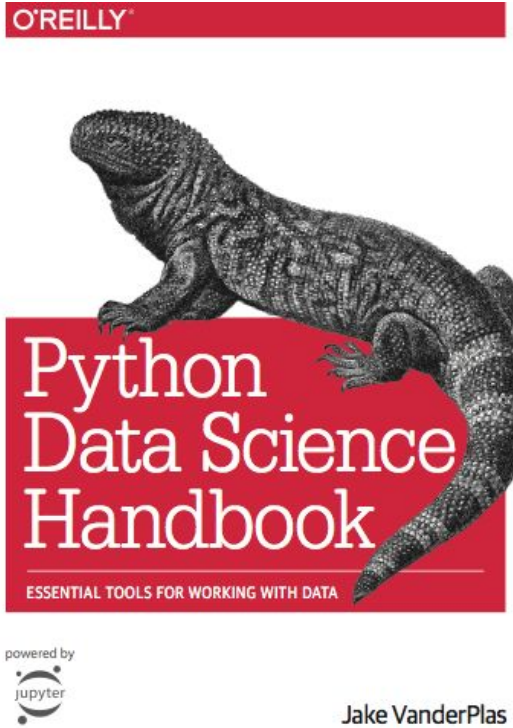
El manual de ciencia de datos en Python
https://jakevdp.github.io/PythonDataScienceHandbook

Es interactivo y lo podés ejecutar en tu máquina.

Pero … es complicado de instalar las dependencias, incluso según el autor

```
$ docker run --rm -p 8888:8888
gmiretti/python-data-science-handbook
```

code: github.com/gmiretti/PythonDataScienceHandbook



Lectura complementaria

Kernels en Kaggle sobre limpieza de datos
https://www.kaggle.com/rtatman/data-cleaning-challenge-handling-missing-values

Son interactivos y lo podés ejecutar en el browser y descargar a tu máquina.

Estan implementados con Docker :D

http://blog.kaggle.com/2016/02/05/how-to-get-started-with-data-science-in-containers/

Solución a la Crisis de Reproducibilidad

- Disponibilidad de los datos crudos ✓
- Código y documentación para repetir los análisis ✓
- Capacidad de analizar correctamente los datos ✓

http://www.pnas.org/content/112/6/1645.full

Laboratorio

Limpiar un set de datos con pandas

- Un set de datos que tengan permisos para compartir con nosotros
  - Más de 100000 registros
  - Más de 20 columnas
  - Con datos con cadenas, números, fechas, y categorías
- Usar los permisos de edificación de San Francisco

https://www.kaggle.com/aparnashastry/building-permit-applications-data/data

El análisis tiene que ser reproducible en las máquinas de los profes

Recomendación: Meetup de Data Science Córdoba

meetup.com/Encuentros-Data-Science-Cordoba/

Conclusiones

Recomendaciones Generales

- Aceptar que la computación es un componente integral de la investigación
- Proveer acceso a los datos crudos primarios
- Registrar las versiones de los conjuntos de datos auxiliares (o archivarlos)
- Guardar las versiones exactas del software usado
- Identificar los parámetros de ejecuciones, incluidos los valores por defecto y otros no evidentes (por ej. semillas de números pseudo-aleatorios)
- Alinear la Ciencia de Datos con las prácticas de Ingeniería de Software

Ejemplo Integrador

Data Science + Software pipelines con Python y Docker
https://github.com/gmiretti/DataScienceExamples

FIN